07-06-00

A/prov

EXPRESS MAIL CERTIFICATE

Date 7/5/00 Label No. EL 6[illegible] 391

I hereby certify that, on the date indicated above I deposited this paper or fee with the U.S. Postal Service and that it was addressed for delivery to the Commissioner of Patents & Trademarks, Washington, DC 20231 by "Express Mail Post Office to Addressee" service.

A. DiLullo A. DiLullo

Name (Print) Signature

PLEASE CHARGE ANY DEFICIENCY UP TO $300.00 OR CREDIT ANY EXCESS IN FUTURE FEES DUE WITH RESPECT TO THIS APPLICATION TO OUR DEPOSIT ACCOUNT NO. 04-0100

DARBY & DARBY P.C.

805 Third Avenue

New York, New York 10022

212-527-7700

File No: **6727/0H381**

Date: July 5, 2000

Hon. Commissioner of
Patents and Trademarks
Washington, DC 20231

Sir:

Enclosed please find a provisional application for United States patent as identified below:

<u>Inventor/s</u> (<u>ALL</u> inventors, including <u>NAME</u>, plus city and state of <u>RESIDENCE</u> for each):

Julian SATRAN - Haifa, ISRAEL

Kalman METH - Haifa, ISRAEL

<u>Title</u>: IMPROVED METHOD TO DETECT END OF RDMA TRANSFER

including the items indicated:

1. Specification.

2. [] Drawings, _ sheet (Fig.)

PROVISIONAL PATENT APPLICATION COVER SHEET

BEST AVAILABLE COPY

Exhibit D

3. [] Assignment for recording to:

4. [] Verified Statement Claiming Small Entity Status

5. [X] Check in the amount of $150.00 filing; $ recording)

Respectfully submitted,

Dated: July 5, 2000

Walt Thomas Zielinski, Esq.
Reg. No. 18,902
Attorney for Applicant(s)

DARBY & DARBY P.C.
805 Third Avenue
New York, New York 10022
212-527-7700

Invention Disclosure:

Improved Method to Detect End of RDMA Transfer

Authors:
Julian Satran
Kalman Meth

Statement of Problem:

RDMA refers to a Remote DMA (Direct Memory Access) feature that is provided on some communications infrastructures. The sender of data specifies, in a form understood by the receiver, where the data should be placed at the receiving end; the application on the receiving end might then place the data without having to examine a complex context, or might even delegate the data placement to specialized hardware. When data has been successfully delivered into the receiver's buffers, the receiver must be notified of the completed transfer (usually be some kind of interrupt mechanism).

There is sometimes a problem, however, to determine when a data transfer has completed, especially if a large data transfer (which we will call a transaction) has been broken into several smaller data transfers (which we will call packets). It would be desirable to inform the receiver that the entire transaction (large data transfer) has been completed, without interrupting (disturbing) the receiver when only some packet (partial data transfer) has completed. An RDMA engine may know how much data has been transferred on each packet (small data transfer), and it may know how much data makes up the entire transaction. The RDMA engine would then have to keep track of how much data has arrived for each pending transaction (large data transfer), and would generate an interrupt when it has received the total number of bytes that were specified for a particular transaction (after having received some number of packets).

The problem is compounded by allowing the transaction (large data transfer) to be broken into several smaller pieces (packets) that may traverse different network fabrics. In this case, no single RDMA engine on the receiving end receives all of the data for a particular transaction (large data transfer), and therefore no single RDMA engine can know when the transaction (large data transfer) has completed. In current state-of-the-art RDMA proposals/implementations, this is solved by generating an interrupt or callback for each packet (small data transfer) on each of the RDMA engines, and computing the total data delivered for the transaction in software. This solution has the undesirable condition that it results in an interrupt being generated for each packet (small data transfer). The receiver is interested in knowing when the entire transaction (large data transfer) has completed, and all of the extra interrupts/callbacks for the small data transfers consume resources that could otherwise be used for other purposes.

It is therefore desirable to have a solution to this problem that minimizes the number of interrupts in determining when a transaction (large data transfer) using RDMA has completed.

Claim:

We propose a method that reduces the number of interrupts. Each RDMA engine that receives some data for a transaction will produce one (and only one) interrupt/callback for that transaction. Any RDMA engine that did not receive any data for a transaction will not produce any interrupts for that transaction.

Solution/Embodiment:

The sender may send parts of data (packets) of the transaction over several network fabrics/connections. When the sender has sent the last packet of data through a particular network, the sender will mark the end-of data through a marker that can be a flag (in the message header) that indicates that this is the last piece of data being sent on this network connection/fabric for the particular transaction or an specially formatted message (e.g. an empty RDMA packet). If the sender finished sending out data for a transaction, but it had sent data earlier over a network without marking the last packet sent on that network, the sender must send a specially formatted message (e.g.. an empty RDMA packet) that marks it as the last packet being sent over that network for the particular transaction. Each receiver thus knows which packet is the last packet it will receive for a particular transaction. Upon receiving this last packet, the RDMA engine generates an interrupt/callback, informing the receiver how much data it has received on its network connection/fabric for the particular transaction. The receiver then keeps track of the sum of data that arrived on each of the network connections/fabrics that reported data received for the particular transaction. When the total number of bytes for the transaction has been received via the various RDMA engines, the receiver knows that the transaction (large data transfer) has completed. Any RDMA engine that did not process any packets for a transaction will not have generated an interrupt for that transaction. Any RDMA engine that did process packets for a transaction will have generated a single interrupt for the transaction after it had processed all of the packets that are to arrive on its network connection/fabric. We have thus reduced the number of interrupts needed to determine when a split RDMA transfer has been completed.
A variant of this technique may involve the information sender to inform the receiver about the connections on which it has sent data enabling him to cross-check the validity of the receive-counts.